面向数据演化的实体解析述评

高广尚[1,2]

（1.中国科学院文献情报中心，北京 100190；2.中国科学院大学，北京 100190）

摘要：分析数据演化下的基于相关性聚类的增量实体解析机制。针对增量实体解析过程展开分析和研究，首先探讨静态数据集中的实体解析、然后分析基于相关性聚类的解析机制，最后研究面向数据演化的实体解析过程。基于相关性聚类的增量实体解析技术能很好地运用于频繁更新的数据环境中。仅从聚类技术角度分析了面向数据演化的增量实体解析技术现状，未给出该技术的详细算法描述。有助于较全面系统地理解面向数据演化的实体解析过程及其内在的相关技术难点。

关键词：数据演化；相关性聚类；增量实体解析

中图分类号：TP393

A Review on Entity Resolution Based on Data Evolution

Gao Guangshang[1,2]

（1. National Science Library of CAS, Beijing 100190）

（2. University of Chinese Academy of Sciences, Beijing 100190）

Abstract: To analyse the mechanism of incremental entity resolution based on correlation clustering on data evolution. This paper analyses and studies the process of entity resolution, first discusses the entity resolution of static data collection, then analyses the mechanism of entity resolution based on Correlation Clustering, finally studies the process of entity evolution on data evolution. Incremental entity resolution based on Correlation Clustering can be used in frequently updated data environment. Only studies the technical situation of incremental entity evolution on data evolution in terms of clustering technique, not gives a detailed arithmetic statements of the technology. This paper will facilitate to give an comprehensive overview of the process of entity resolution on data evolution and its inherent technical difficulties.

Keywords: Data Evolution; Correlation Clustering; Incremental Entity Resolution

1 引言

实体解析就是识别出数据集中那些表示同一现实世界实体的记录的过程。事实上，实体解析过程不是一次性的，而是随着数据、模式和应用不断变化而变化的过程[1]。大数据时代背景下，越来越多的数据在进一步被分析处理前需要匹配、聚类或整合，因而面向数据演化的实体解析研究受到越来越多的关注[2]。然而，这一研究由于数据具有海量、多样、异构和动态变化等特性而受到了新的挑战。主要体现在以下两个方面：首先，解析过程因为数据量大而花费较长的解析时间，如数据集中包含数以亿计的记录。其次，解析结果因为数据的快速更新而很快变得过时、无效，如记录的属性值随时间而更改。为有效应对这一挑战，面向数据演化的实体解析机制不仅要有与传统朴素方法（从头开始对数据集执行实体解析）类似的查准率和查全率，而且要有比传统朴素方法更快的解析效率。

尽管面向数据演化的实体解析主要解决如何保持解析结果最新这一问题，但它需要基于一个已清洗、整理过的“干净”数据集，因为需要利用数据集中已识

别出的聚类结果[3]。这样，在后续的数据演化下的解析过程中可避免大量重复性工作（如不必要的记录对比较），从而满足近乎实时的解析需求。本文在回顾相关研究的基础上，从对数据进行聚类的角度阐述面向数据演化的实体解析机制。为此，笔者主要从相关研究、静态数据集中的实体解析、基于相关性聚类（Correlation Clustering）的解析机制和面向数据演化的实体解析 4 个方面进行梳理和分析，以期为未来进一步的相关研究提供基础。

2 相关研究

为有效解决数据演化下的增量实体解析问题，现有大部分研究主要是基于聚类算法来设计一种增量聚类算法，从而能充分利用先前的聚类结果以对新到数据实施有效的解析。为此，笔者从经典聚类算法和一般聚类算法两方面来综述基于它们的增量实体解析研究。

2.1 基于经典聚类算法的增量实体解析

在增量实体解析研究中采用的经典聚类算法主要包括凝聚层次聚类算法、K-menas 算法和相关性聚类算法。它们共同的优点就是算法思想简单，适合于大量数据环境下的聚类。

2.1.1 凝聚层次聚类算法

凝聚层次聚类算法满足增量性质，因而基于它设计的增量聚类算法能满足数据演化下的实体解析需求。

针对动态环境中非增量层次聚类方法面临的效率低下问题，DH Widyantoro 等[4]提出了增量凝聚层次聚类算法（Incremental Hierarchical Clustering，IHC），旨在构建一个满足同质性（Homogeneity）和单调性（Monotonicity）的层次结构。同质性簇是一个有相似密度的对象的集合。如果一个簇的密度总是高于其父辈簇，那么簇的层次结构满足单调性。算法以自底向上的方式运行，在将新来实例放置于层次结构后，算法只对受新实例出现所影响的区域进行一系列层次结构调整过程。

为对度量空间（Metric Space）中的动态结点集进行聚类，在受到信息检索领域一些应用的启发下，诸如文档和图像分类等应用，Charikar, M.等[5]提出了基于层次凝聚的增量聚类算法（Incremental Clustering Algorithms，ICA）。算法的目标是，随着新结点的插入，算法能有效地维持一些具有最小直径的簇。涉及的增量聚类问题定义为：对度量空间中一个有n个结点的更新序列（Update Sequence）来说，维持一个有k个簇的集合，使得每当有新结点出现时，它或者分配到当前k个簇中的某个簇，或者在该集合中新增一个包含该结点的簇，此时需要将两个现有的簇合并成一个簇，因为簇的总数预定为k。不同于其他聚类技术，该增量聚类算法需要预先设定簇的总数。

Omar Benjelloun 等[6]将实体解析问题分成了两个方面：匹配与合并记录的黑盒函数（black-box）和调用这些函数的实体解析算法。这种划分带来的两个好处是：产生一些可被许多应用使用、具有良好语义结构的通用实体解析算法；专注于算法性能指标（减少对潜在昂贵的黑盒函数的调用次数）。在此基础上，作者提出了一个很容易适应新数据或特征不断出现的增量环境下的“F-Swoosh”算法。由于算法利用多个哈希表（Hash Tables）来保存值，因此，当新记录出现时，可不必在整个记录集上运行“F-Swoosh”算法，从而避免在记录间进行一些不必要的比较，尤其是在已经知道记录不匹配的情况下。对新特征的处理与此类似。

2.1.2 K-means 聚类算法

针对 K-means 聚类算法倾向收敛于局部最优的问题，Pham 等[7]提出了一种通过移动簇中心以减少簇失真（Cluster Distortion）的增量 K-means 聚类算法（Incremental K-means）。提出的搜索策略减少了算法对簇中心初始化的依赖，并且算法仅需要运行一次就能实现几乎最佳的结果。

2.1.3 相关性聚类算法

针对数据项以在线方式出现的在线聚类问题，Claire mathieu 等[8]研究基于相关性聚类的增量相关性聚类算法（Incremental Correlation Clustering），其主要关注两点：①每次加入一个顶点；②已识别的聚类结果需要保存。当数据项v一到达，v和先前到达的数据项之间的关系就会被揭示，结果是对于每个数据项u，我们知道它是否与v相似。算法可能会为v产生一个新的簇，并将它与现有的簇合并。由于算法一直维持数据项的聚类结果到相似的类别，因而非常适合那些对某事感兴趣的应用。

针对大数据时代下的数据更新快而使先前解析结果很快失效的问题，例如，某些属性值随时间变化而变化等。Anja Gruenheid 等[9]提出了一个端到端框架来对增量操作（包括插入，删除和修改操作）涉及的记录实施增量聚类。其中基于相关性聚类的增量聚类算法，不仅能增量更新聚类结果，而且能在不影响原有聚类结果的情况下将增量操作涉及的记录与现有的簇进行合并或分离，并能利用增量操作中的新证据来修正先前存在的聚类错误情况。重要的是，算法能显著减少聚类过程所需要的时间，同时无损聚类质量，进而满足面向数据演化的近乎实时的解析需求。

2.2 基于一般聚类算法的增量实体解析

由于增量实体解析与聚类数据流（Data Streams）的问题密切相关，因而可利用研究聚类数据流的方法来研究增量实体解析问题。

针对动态数据集中随时存在增加、删除记录的可能这一问题，Can 等[10]提出了适用于动态信息处理的增量聚类算法，该算法能在不显著影响当前所有簇的情况下，只分析与改变相关的簇。

Aggarwal 等[11]认为针对数据流开发的聚类算法，尽管解决了聚类技术中的可扩展性问题，但通常对数据演化问题视而不见，并且没有解决以下 2 个问题：（1）当数据随时间不断演化时，形成的簇的质量差。（2）数据流聚类算法需要更多的功能以在数据流的不同部分上去发现和探索簇。为了聚类大量演化的数据流，作者提出了一个高效率的 CluStream 算法。该算法有超越其他技术的明显优势是：其他技术试图一次性聚类整个数据流，而不是将数据流看成是一个随时间不断改变的过程。CluStream 算法能描述（Characterize）演化环境中不同时间段上的数据流簇。

Steven Euijong Whang 等[1]对现有不能适应数据演化的聚类技术进行改进，定义了一个能保证数据演化准确性的增量性质（General Incremental，GI），提出了一个满足该增量性质的增量数据算法（Incremental Data Algorithm）。由于能充分利用先前的聚类结果，因此，增量数据算法在面向数据演化时不仅能一次解析一条记录，而且有较好的解析效率。

Heiko Müller 等[3]认为清洗数据是一项耗时且代价高昂的任务。在已执行数据清洗并获得一个无错误的“干净”数据集后，当数据集中的一些记录值出现变

更时，人们不想完全执行整个数据清洗过程。仅对受变更值影响的部分重新执行清洗过程即可，其中，通过分析清洗谱系（Cleansing Lineage）来确定受影响的部分。清洗谱系不仅维持那些已正确识别的记录，而且维持那些在清洗过程中证实为正确识别的记录。

Hernandez 等[12]认为一旦数据集已被清洗（通过实体解析方法）并存储以供将来使用，那么在重新运用清洗方法前，将已清洗过的数据与新到来的数据进行串接（Concatenation）可能不是要遵循的最佳策略。特别地，新的增量数据在短期内可用的情况下，在对数据进行清洗前，串接所有数据被证明是在时间和空间方面代价高昂。为此，作者提出了一个增量式清洗算法，在短时间内能很好地解析新增数据。

3 静态数据集中的实体解析

在静态数据集（数据集是静态不变的）上的实体解析通常采用索引技术（Indexing Techniques）来减少需要比较的记录对[13]，或者说过滤掉那些非常不可能匹配的记录对，进而让比较只在那些最有可能匹配的记录对间进行，最终让解析速度加快。本质上，索引技术会将整个数据集划分成多个块，那些位于块内的记录称为候选记录，它们将会被详细比较（计算候选记录对的详细相似性值）。对于大部分只包含一条记录的块来说，其中的记录被认为已识别出来，即它们单独表示一个现实世界实体，因而无须再进行详细比较。很显然，对块中包含的多条记录进行详细分析，以识别它们是否表示同一个现实世界实体，是静态数据集中实体解析的研究重点。鉴于此，笔者从计算相似性和构造相似图形两个方面进行分析。

3.1 计算相似性

在决定两条记录是否表示同一个现实世界实体时，需要将两条记录对应属性上的值进行逐一比较，并计算出相似性值[14]。计算过程如图 1 所示，假设比较的两条记录r，r'同属于人物类，属性为名字、邮箱等，方框中的值表示记录对应属性上的值，方框间连线上的实数值（如 0.8）表示对应属性的值的相似性大小。很显然，这里需要一个阈值（α）来判定对应属性的值的相似性是否成立，即如果相似性值大于该阈值时，就认为对应属性彼此相似。在计算出对应属性的相似性值后，接下来，仅将那些相似性成立的属性的相似性值进行线性相加，就可计算出两条记录间的相似性值。具体可通过式子$s(r,r') = \sum f(r,r')$来计算，其中，s表示用来计算两条记录是否相似的相似函数，或称为记录分类器，f表示用来计算两个对应属性是否相似的相似函数，或称为属性分类器[15]。同样，这里需要一个阈值（β）来指定两条记录的相似性是否成立。如果两条记录的相似性值大于该阈值，即$s(r,r') > \beta$，那么就认为两条记录r和r'表示同一个现实世界实体。这种判定两条记录是否相似的方法称为成对相似性方法（Pairwise Similarity）[16-18]。

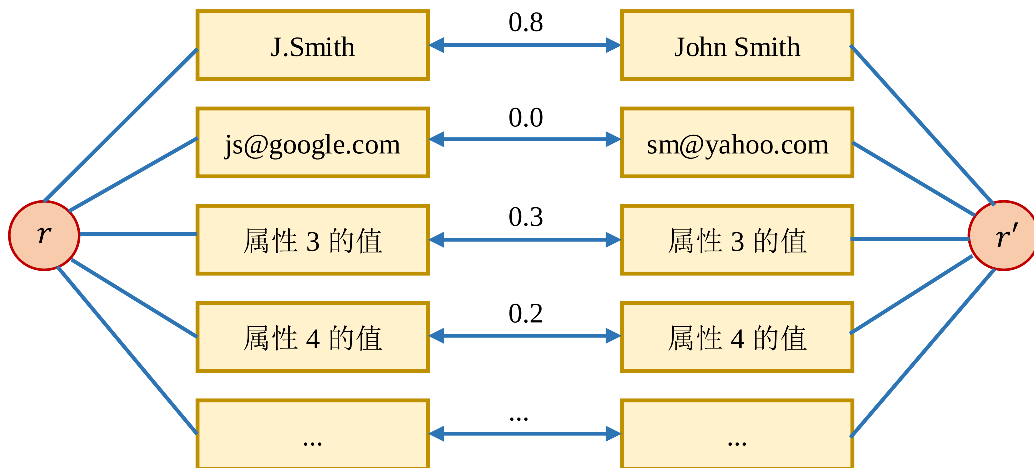


图 1 计算记录间的相似性（Pairwise Similarity）

值得注意的是，记录间的相似性计算过程与具体的应用环境有关，主要涉及 3 个方面：一是采用哪种相似性度量来对属性值进行比较，如编辑距离、字符串精确匹配、Ad hoc 相似和 Jaccard 相似等[15]；二是如何组合这些计算出的相似性值，如采用一般线性相加、考虑属性的权重[19]；三是如何设定作为判定标准的阈值α、β的大小[20]。

3.2 构造相似图形

为系统地阐述静态数据集中的实体解析过程，这里不考虑能加快解析速度的索引技术。一旦确定了如何计算记录间相似性的方法，接下来就可对单个数据集中的所有记录对计算相似性值（如同对块内的记录对进行相似性计算，以判定它们是否表示同一个现实世界实体）。在单个数据集中逐对计算记录间是否相似的过程，实际上就是对表本身中的记录做笛卡尔积[21]，即比较所有可能的记录对。对两个不同的数据集而言，计算过程与此类似。表 1 中的数据集包含 6 条记录（$r_1 - r_6$），按照笛卡尔积的逐对比较方式，共需进行 6(6-1)/2=15 次比较。

表 1 数据集

记录 ID	名	姓	年龄	街道名	城郊	出生日	出生月	出生年
r_1	john	smith	18	miller st	dickson	12	11	1970
r_2	jonny	smith	73	miller st	dixon	11	10	1970
r_3	joan	smith	73	dawson cr	lyneham	11	12	1979
r_4	max	miller	73	dawson cr	lyneham	11	2	1969
r_5	sal	bass	67	milles rd	ainslie	28	5	1981
r_6	sally	bass	64	miles rd	ainsile	23	5	1981

表 2 中列出了这些记录间的相似性值（通过图 1 中所示的方式计算），以及它们与阈值比较后的匹配状态，其中，匹配状态的判定过程称为成对分类技术[22]，即相似性值大于阈值 5.0 就匹配，否则不匹配。表 2 中共有 4 对记录是匹配的。

表 2 数据记录匹配表

候选记录对	相似性值（函数 SimSum 算出）	Classification（阈值β=5.0）
(r_1, r_2)	5.20	**匹配**
(r_1, r_3)	3.30	不匹配
(r_1, r_4)	1.15	不匹配
(r_1, r_5)	0	不匹配
(r_1, r_6)	0	不匹配

(r_2, r_3)	5.05	**匹配**
(r_2, r_4)	2.70	不匹配
(r_2, r_5)	0	不匹配
(r_2, r_6)	0	不匹配
(r_3, r_4)	5.25	**匹配**
(r_3, r_5)	0	不匹配
(r_3, r_6)	0	不匹配
(r_4, r_5)	0	不匹配
(r_4, r_6)	0	不匹配
(r_5, r_6)	6.20	**匹配**

然而，这种单独而非集体地将记录对划分成匹配或不匹配的决定将会引发判定不一致性矛盾，即传递闭包问题[23]。为方便分析这种问题的本质，可依据表 2 中记录对的相似性值（匹配信息）构造出一个相似图形（Similarity Graph）[24]，用符号G来表示，如图 2 所示。图 2 一个结点表示一条记录，结点间的实线边表示两条记录相似，实线边上的数值表示相似性值（如 5.25），或称为边的权值。4 条实线边表示存在 4 个匹配。

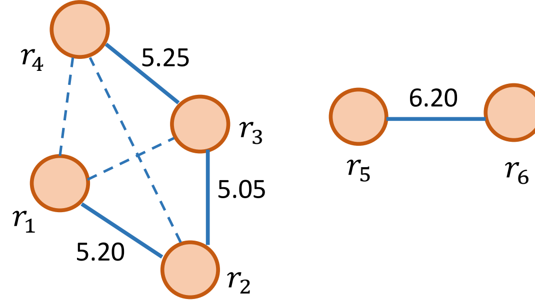


图 2 构造的相似图形

传递闭包是指这样一种情况：如果记录对(r_i, r_j)被划分为匹配（表示同一个现实世界实体），而且记录对(r_j, r_k) 被划分为匹配，从而记录对(r_i, r_k)匹配，但是记录对(r_i, r_k)在计算相似性值时却被划分为不匹配。这与直觉相矛盾，图 2 中记录对(r_1, r_2)、(r_2, r_3)和(r_3, r_4)匹配（实线相连），由于匹配的传递性，那么记录对(r_1, r_4)也应是匹配的（实线连接的结点应被看成一组，表示同一个现实世界实体），但在计算记录对(r_1, r_4)时却被划分为不匹配（如表 2 所示）。很显然，如果应用传递闭包这一性质，那么将会涉及改变记录对的匹配状态，使得组内的记录间不会出现匹配状态相矛盾的情况[25]。即尽管结点彼此相连，但它们不应划分到同一组内，这样，划分到同一组内的结点就将满足传递闭包这一性质。

另外，匹配的传递性也将导致记录“链”问题。图 2 中包含两条记录“链”，分别是$r_1 - r_4$组成的记录“链”和$r_5 - r_6$组成的记录“链”。然而，在“链”两端的记录彼此可能完全不同，因而它们不应被认为是匹配的。如记录对(r_1, r_4)的相似性值为 1.15，因而它们不可能表示同一个现实世界实体。

尽管有研究认为：在现实数据集中，通过成对分类技术产生的记录“链”问题似乎很少发生，因为在不同记录属性中的所有可能值的空间非常大，因此，那些不匹配记录对彼此具有很高相似性的可能性非常少[23, 26, 27]。

但是，为解决匹配记录“链”问题，现有的研究主要采用聚类方法（如相关性聚类）来将一组记录划分为匹配，而不仅是通过单独记录对匹配的方式来将一组记录划分为匹配。

4 基于相关性聚类的解析机制

为解决记录“链”问题，相关性聚类方法将通过对簇内结点相似性和簇间结点相似性的分析来综合决定，“链”上的哪些记录应划分到一起形成簇可能会更合理，而不是简单地将“链”上的所有记录都划分到同一个簇中。从而让最终形成的簇满足传递闭包要求，即簇内的记录彼此都相似且表示同一个现实世界实体。

为重点而详细地梳理相关性聚类解析机制，笔者主要从相关性聚类的基本思想和目标函数两个方面进行分析。

4.1 基本思想

相关性聚类被认为是实体解析的一个标准方法，最初由 Bansal 等人提出[28]，其算法的输入与其他聚类算法相同，即算法的输入都是相似图形G，其中，图中的每个结点代表数据集中的一条记录，每条边代表连接的结点相似，边上的权值为结点间的相似性值$sim(r,r')$ （$0 \le sim(r,r') \le 1$）。图 2 就是这样一个相似图形。相关性聚类就是在相似图形上进行自动划分，即对图中结点进行聚类。

这种基于图划分所进行的聚类并不是孤立地基于记录对间的相似性值进行识别决策，而是充分考虑了多个待识别结点之间的相似性。因此，当确定将图中某个结点分配到某个簇时，不是简单地依赖该结点与某一个其它结点的相似性值，而是依赖于该结点与这个目标簇中的所有结点，甚至其他簇中所有结点的相似性值。

此外，与其他聚类算法相比，例如 k-Means 聚类算法等[29]，基于相关性聚类的实体解析存在 5 方面的优势：其一，有清晰的聚类质量概念，即形成的聚类结果满足“一致性权值最大”或“不一致性权值最小”原则[28]；其二，不需要预先指定聚类结果中簇的数目，该数目可以是 1 到n之间的任何一个数；其三，不依赖数据出现的顺序，并且以无监督方式学习聚类；其四，相似图形中边的权值可以是位于[0,1]区间内的任意实数值，不一定是 0 或 1 这种二元值。其五，相关性聚类代表一种基于邻接（Adjacency-based）测量的图形聚类方法[30]，因此，适合在相似图形上进行聚类。

最后，相关性聚类的目标是在相似图形上找到一个最佳聚类结果，使得该聚类结果尽可能与结点间的相似性值（边的权值）一致。

4.2 目标函数

4.2.1 相关定义

相关性聚类在相似图形上找最佳聚类结果的过程，实质上是一个整数规划问题，而且是一个 NP-hard 问题，因而需要采用近似求解方法[28]。尽管有两种采用不同近似观点的策略去解决这个问题，但解决这个问题的关键是如何定义一个目标函数（Objective Function），因为相关性聚类利用目标函数（惩罚函数）来评价聚类结果的质量，并选择能优化目标函数值的某个聚类结果作为最佳聚类结果。为便于清晰地说明目标函数是如何被定义的，笔者仅从聚类结果满足“不一致性权值最小”原则这一策略进行分析。

与其他依据边的权值来删除边、结点的方法类似[25]，目标函数的定义也是基于边的权值。不同于文献[28]在定义目标函数时只考虑边的权值为二元值的情况：“1”表示相似，“−1”表示不相似，文献[9]将边的权值定义为位于某区间段（如[0.8-1.0]）的任意实数，因而基于这样的权值定义的目标函数也更切合实际。不管权值怎样定义，为使聚类结果满足“不一致性权值最小”原则，定义目标函数

的思想是：奖励簇内的高内聚性（High Cohesion），惩罚簇间的高相关性（High Correlation）。其中，高内聚性是通过簇内结点间的相似性或紧密度测量来判定，高相关性是通过簇间结点间的相似性或紧密度测量来判定。

具体来说，为了定义这样一个目标函数以实现近似求解，主要从两方面来进行考虑：对簇内的每对节点来说，存在一个关于它们不相似的聚合惩罚（Cohesion Penalty），值为$1-sim(r,r')$；对簇间的每对节点来说，存在一个关于它们相似的相关性惩罚（Correlation Penalty），值为$sim(r,r')$。接下来，就可基于这些惩罚值定义出一个满足要求的目标函数，如式（1）所示。很显然，如果通过式（1）计算出的总惩罚值最小，那么其对应的聚类结果将是满足“不一致性权值最小”原则的最佳聚类结果，同时意味着簇内的结点将保持高聚合性，簇间的结点将保持低相关性。

$$CC(\mathcal{L}_G)=\sum_{C\in\mathcal{L}_G,r,r'\in C}(1-sim(r,r'))+\sum_{C,C'\in\mathcal{L}_G,C\neq C',r\in C,r'\in C'}sim(r,r') \quad (1)$$

其中，$C,C'\in\mathcal{L}_G$表示，C,C'是聚类结果$\mathcal{L}_G$（$\mathcal{L}_G$表示G上的聚类结果）中的两个不同簇。$r,r'\in C$表示，r,r'是簇C内的两个不同结点。$r\in C,r'\in C'$表示，r,r'分别属于两个不同的簇C和C'。

4.2.2 实例分析

为深入分析相关性聚类是如何通过计算目标函数来选择一个最佳聚类结果，笔者通过一个启发性例子来详细阐述这一过程。

图 4（a）是一个相似图形，通过对 10 条记录r_1- r_{10}进行成对计算后得到，计算方式如表 2 所显示的那样，例如r_1和r_2之间的相似性值为 0.9。另外，如果任意记录对的相似性值小于等于 0.7，那么它们间的边将被省略掉。假设相关性聚类在图 4（a）的相似图形上进行一次聚类运算后，得到如图 4（b）所示的聚类结果，包含 5 个簇C_1-C_5。接下来，用目标函数来判定该聚类结果是否是最佳的，即对簇内结点、簇间结点的惩罚情况进行计算分析。

对簇C_1有一个聚合惩罚，总值为 0.2+0.2+0.1+0.1+1=1.6，其中，r_2和r_4之间的值为 1-0.8=0.2，r_3和r_4之间的值为 1-0.8=0.2，r_2和r_1之间的值为 1-0.9=0.1，r_3和r_1之间的值为 1-0.9=0.1，r_4和r_1之间的值为 1-0=1。

对簇C_4有一个聚合惩罚，总值为 0.2+0.2+0=0.4，其中，r_9和r_8之间的值为 1-0.8=0.2，r_9和r_7之间的值为 1-0.8=0.2，r_8和r_7之间的值为 1-1=0。

此外，聚类结果对簇C_4和C_5间有一个相似性惩罚（因为有边相连），值为：0.8，其中该值为r_9和r_{10}间的相似性值。

需要强调的是，C_2和C_3是单点簇，即该簇仅由一个单独的结点组成，而且它们也不与其他的簇相连，因此对它们不存在相关的惩罚。

最后，通过目标函数计算出的总惩罚值为：$CC(\mathcal{L}_G)=1.6+0.4+0.8=2.8$。事实上，这个总惩罚值 2.8 是在所有可能聚类结果中算出的总惩罚值中的一个最小总惩罚值（鉴于计算过程的复杂性，这里没有给出如何计算出另一个聚类结果的总惩罚值，并用它来进行比较）。因此，该总惩罚值所对应的聚类结果将满足“不一致性权值最小”原则，因而其被认为是最佳的聚类结果。

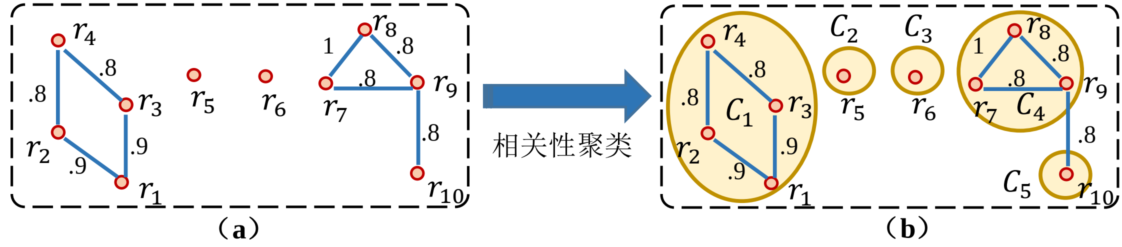


图 4 相关性聚类结果

与通过匹配的传递性来简单判定相连的一组记录彼此匹配的情况不同，相关性聚类将通过目标函数来进一步分析它们的匹配情况，以获得最佳的聚类结果，如将连在一起的r_7- r_{10}划分成两个簇：r_7- r_9（C_4）和r_{10}（C_5），而不是原先的一个簇。

5 面向数据演化的实体解析

通过上面的分析，我们了解到相关性聚类旨在在相似图形G上找到一个最佳的聚类结果。然而，在实际应用中，数据集在每段时间都会有新的数据新增、删除或修改（统称为增量操作），这些操作将会相应地在原来相似图形上带来一些变化ΔG。在理想情况下，我们仍希望能在相似图形$G + \Delta G$上找到一个最佳的聚类结果，从而实现面向数据演化的实体解析目标。

为较全面系统地分析面向数据演化的实体解析过程涉及到的关键技术，笔者从增量聚类过程、增量操作、增量性质和增量聚类算法四个方面进行详细阐述。

5.1 增量聚类过程

对演化的数据进行解析的过程本质上是一个增量聚类过程。增量聚类过程主要分为三个步骤：记录对的相似性计算、基于相似图形的相关性聚类和面向数据演化的增量聚类。图 5 中用序号①、②和③分别对各个步骤进行了标注。从图中可以看出，第三步面向数据演化的增量聚类的实现是在第二步聚类结果的基础上，对增量操作涉及的记录进行聚类分析。很显然，这个聚类分析过程具有一定的挑战性，因为它需要解决如何对原有簇的结构进行调整，调整的依据是什么，以及如何利用新到来的证据修正先前聚类结果中可能存在的错误等。为对增量聚类过程的完整性有一定的了解，笔者将简明扼要地阐述其中的三个步骤及其关联性。

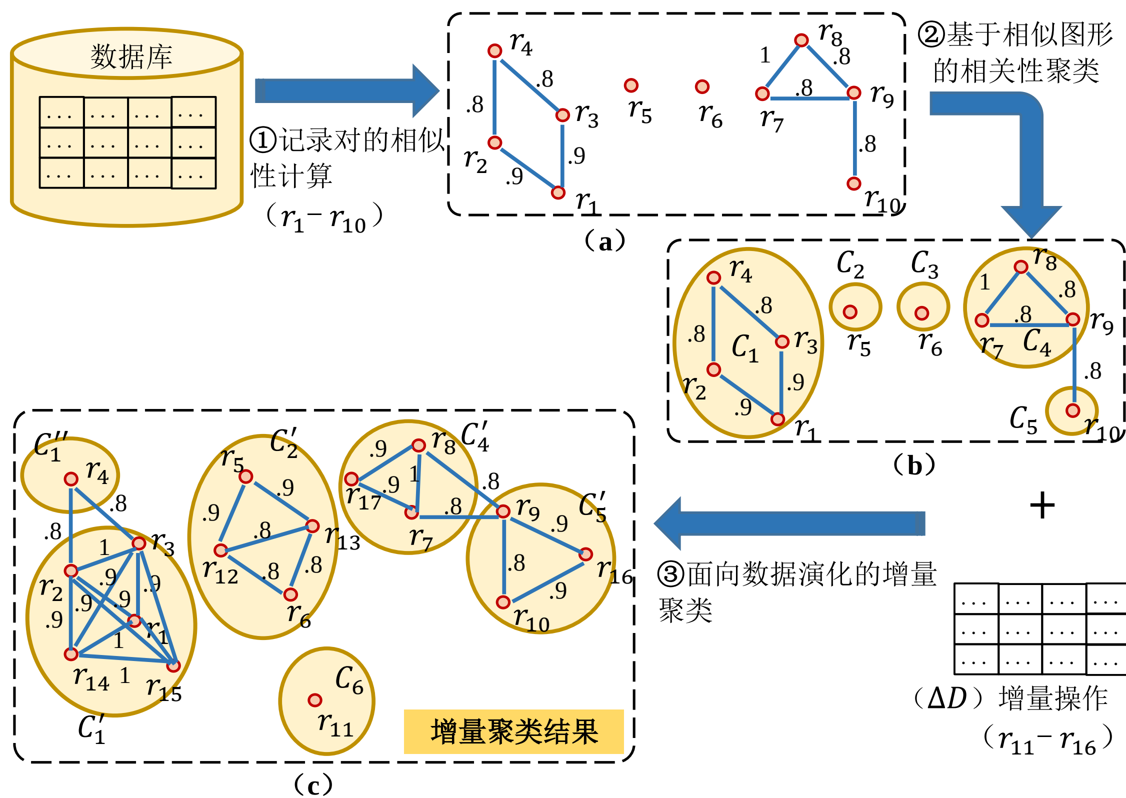


图 5 增量聚类过程

第一步，记录对的相似性计算：计算数据集（r_1- r_{10}）中记录对的相似性值，并将那些大于阈值（如 0.7）的相似性值作为边的权值来构造一个带权的相似图形G，如图 5（a）所示。这时，如果简单地按照匹配的传递性来分析它们，即将相连的结点看成是一个簇（组、块），那么将形成 4 个簇：r_1- r_4、r_5、r_6和r_7- r_{10}。每个簇代表一个现实世界实体。本质上，每个簇是一个记录子集，簇内的记录语法上彼此相似。然而，这种分析可能会导致出现错误，即尽管它们相连在一起，但未必是属于同一个簇。

第二步，基于相似图形的相关性聚类：针对第一步中对簇形成的分析可能存在错误，相关性聚类能在相似图形上依据目标函数来计算并形成簇，而不是依据匹配的传递性。具体来说，相关性聚类的聚类算法以相似图形作为输入，依据目标函数来计算出一个最佳的聚类结果，这意味着尽管有些结点相互连在一起，但它们被划分到不同的簇可能会更好些。如图 5（b）中，经过相关性聚类分析后，r_{10}被单独划分成是一个簇C_5，这样，聚类结果包含 5 个簇：r_1- r_4（C_1）、r_5（C_2）、r_6（C_3）、r_7- r_9（C_4）和r_{10}（C_5），比第一步多了一个簇。

第三步，面向数据演化的增量聚类：与前面在与静态数据集对应的相似图形上进行聚类分析不同，面向数据演化的增量聚类解决的是如何对增量操作（ΔD，r_{11}- r_{16}）中涉及的记录进行聚类分析。很显然，增量聚类不仅需要将增量操作中涉及的每条记录看成是一个单点簇（簇内仅包含记录本身），而且需要利用先前的聚类结果，这样，才能像凝聚聚类（如 Swoosh[31]）那样去迭代地合并相似簇，从而实现增量聚类目标。具体来说，当增量操作（ΔD，r_{11}- r_{16}）依次到达时，其涉及的每条记录将被看成是一个单点簇，将其与先前聚类结果中的簇（$C_1 - C_5$）进行合并，接着，增量聚类算法就可在所有这些簇上进行聚类分析以得到一个最佳的聚类结果。这种聚类分析不仅会联合考虑结点间的关系，而且会利用新来的

证据去识别并修正先前错误的聚类结果。如图 5（b）中的簇C_2和C_3被合并成 5（c）中的簇C_2'，即它们合并成同一个簇可能会更好。

5.2 增量操作

面向数据演化的增量聚类过程本质上是基于增量操作的增量聚类过程。增量操作ΔD不仅让静态数据集动态变化，而且也让与静态数据集对应的相似图形G动态变化，并形成增量图形ΔG。在形成增量图形ΔG的过程中，增量操作ΔD主要涉及到三种具体的子操作：

（1）新增操作（Insert）：向数据集中插入一条记录相当于在相似图形G中增加一个节点，并增加与该结点相连的多条边。具体来说，首先，在ΔG中包括这个新结点，并找到与新结点关联性强的簇。接着，将新结点与簇内各结点依次进行比较以得到相似性值，如果相似性值大于阈值，那么就将它们间的边（包含作为权值的相似性值）加入到ΔG中。

（2）删除操作（Delete）：在数据集中删除一条记录相当于在相似图形G中删除一个节点，并删除与该节点相连的多条边。具体来说，在ΔG中包括这个删除的结点和与其相连接的边。

（3）更新操作（Change）：更新数据集中一条记录的操作相当于在相似图形G中删除与该结点相连的现有的边，并向相应的结点增加新的边。具体来说，在ΔG中包括权值发生改变的边。

值得注意的是，更新操作是修改现有记录的一个或多个属性值。更新操作可通过首先删除旧的记录，然后插入一条新记录来实现。然而，有时直接考虑更新属性值可能会更有效。

本质上，形成增量图形ΔG的过程，就是在原来的相似图形G上增加点、边的过程，因而可用$G + \Delta G$来表示这一过程的结果。很显然，$G + \Delta G$还是一个相似图形，因为其构成方式与静态数据集下的构成方式一样。

5.3 增量性质

尽管增量操作形成了新的相似图形$G + \Delta G$，但原来的相似图形G上的聚类结果信息依然还在。很显然，如果相关性聚类仍能利用这些先前的聚类结果，那么它就能对增量操作实施增量聚类分析。然而，相关性聚类本身并不支持增量聚类分析，因为它只能在结点层面（一次解析一个结点）而非簇层面上进行分析，从而无法利用先前的聚类结果。为让相关性聚类能在簇层面上进行增量聚类分析，以实现面向数据演化的实体解析目的，必须让相关性聚类（本质上是算法）满足增量性质，其定义如下：

增量性质（GENERAL INCREMENTAL）[1]：定义F为一个批量聚类算法，算法的输入是记录的聚类结果（对应图 5（b））。令$S(G)$是一个由相似图形G中每个结点形成的单点簇所组成的集合。对某个子图$G' \subseteq G$，如果有$F(S(G \backslash G') \cup F(S(G')) = F(S(G))$，那么就说算法$F$满足增量性质。

式子$F(S(G'))$表示，批量聚类算法对子图G'中的单点簇进行聚类。式子$F(S(G \backslash G')$表示，对相似图形G中除子图G'外的单点簇进行聚类。式子$F(S(G))$表示，对相似图形G中的所有单点簇进行聚类。

式子$F(S(G \backslash G') \cup F(S(G')) = F(S(G))$表示，等式两边的聚类结果相同，并且所使用的算法相同，但却采用不同的聚类思想。等式右边采用的是批量聚类思想，即对相似图形G中所有单点簇进行批量聚类分析（迭代地合并相似簇）。等式左边采用的是增量聚类思想，即先将相似图形G图形分成两个不相交的部分 G'和

$G \backslash G'$（可分别可看成是原有的相似图形和增量图形），然后，先对G'进行批量聚类分析，后对$G \backslash G'$进行批量聚类分析，最后将两部分的聚类结果合并形成最终的聚类结果。

当相关性聚类算法满足增量性质时，我们就可将增量图形ΔG中的单点簇与先前聚类结果中的簇合并在一起，并在其上直接运用相关性聚类算法来进行聚类分析。换句话说，我们能基于增量性质来定义增量聚类结果$f(G, \Delta G, \mathcal{L}_G) = F(\mathcal{L}_G \cup S(\Delta G))$。

5.4 增量聚类算法

通过对增量性质的分析可知，为获得增量聚类结果，我们最终只要在相似图形$G + \Delta G$上运行满足增量性质的相关性聚类算法即可。很显然，我们需要一个增量聚类算法来找到对应的子图，并调用相关性聚类算法在该子图上进行聚类分析即可。这样，设计增量聚类算法时的主要思想是：更新与增量操作涉及的记录直接相关联的那个子图，而不是其他子图。

通过对增量操作的分析可知，这个直接相关联的子图要么是原来就存在的相关联子图（在相似图形上删除、修改结点的操作），要么是新近形成的相关联子图（在相似图形上增加结点的操作）。图 6 中“涉及的子图”表示一个与增量操作涉及的记录直接相关联的子图。本质上，子图就是一个由簇组成的集合，包括先前聚类结果中的簇（蓝圆圈表示），以及增量操作中涉及的簇（橙圆圈表示）。

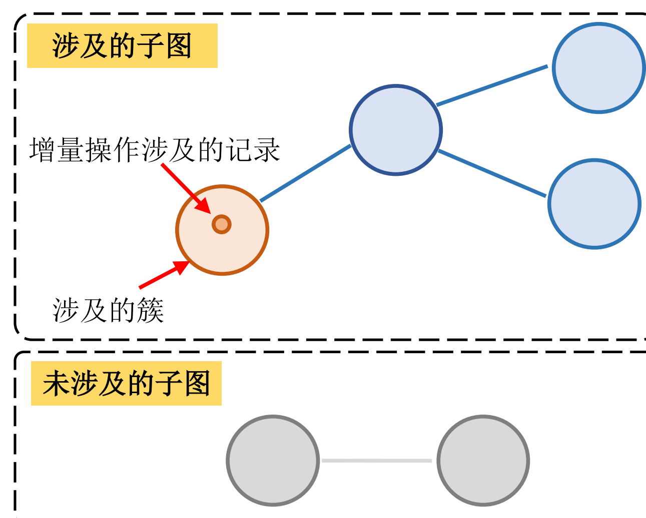


图 6 相似图形中的两个子图

最后，依据设计的思想和直接相关联子图的确定方式，增量聚类算法主要分为三个步骤：

第一步，找到直接相关联的子图。在实施子图查找时要区分是哪种增量操作，例如，删除和修改操作下的查找方式类似，而增加操作下的查找方式与之不同。

第二步，找到直接相关联子图上的最佳聚类结果。在找到的直接相关联子图上使用相关性聚类算法来进行聚类分析，进而得到该子图的新的最佳聚类结果（包含最佳簇）。

第三步，构造出新的最佳聚类结果。通过利用第二步中子图上的最佳簇替换其原来的旧簇，来从原来的最佳聚类结果中构造出新的最佳聚类结果。

6 结论

随着数据日益增长，基于聚类技术的增量实体解析正成为一项越来越重要的任务。相较于其他聚类技术，相关性聚类具有诸多优势，因而基于相关性聚类的增量聚类算法能满足数据演化下的增量解析需求。然而，为有效实现面向数据演化的实体解析目标，设计增量聚类算法时需要考虑两个方面的因素：一是获得的聚类结果质量应该与批量聚类算法相似；二是应该有比批量聚类算法更快的运行速度，特别是在增量操作次数较少的情况下。只有这样，增量聚类算法的聚类质量和聚类效率才有可能得到保障。

本文回顾了增量实体解析的相关研究，并系统深入地梳理和总结了面向数据演化的增量聚类过程，在此基础上，笔者了解到现有研究还存在两点不足：

（1）没有考虑到如何高效快速定位到与增量操作涉及的簇直接相关的子图。例如对于增加记录的操作，在构造增量图形ΔG前，我们必须先从相似图形的众多子图中确定哪个子图最有可能与增加的记录相似，然后再计算它与这个确定子图中各结点间的相似性值。显然，这个查找子图的过程对增量聚类算法的执行速度有着关键的影响，这一问题被看成是相似性连接问题[14]。

（2）没有考虑到如何快速比较两个簇的相似性。在查找子图的过程中，有一些很可能匹配的候选子图，为进一步快速确定哪个子图最有可能，需要对两个簇进行快速比较，这涉及到簇代表问题，因为簇的比较过程实质上是将簇内某个代表记录与另一个簇的代表记录进行比较。显然，计算两个簇之间的相似性值时所采用的方法（如单链接、全链接、平均链接和代表记录等）与具体的应用相关，并且还要仔细考虑应用构建者的意见[19]。

作者简介：高广尚，中国科学院文献情报中心 2013 级博士研究生，gaoguangshang@mail.las.ac.cn；研究方向为：实体解析、数据质量和智能信息处理。

7 参考文献

[1] WHANG S E，GARCIA-MOLINA H．Incremental entity resolution on rules and data [J]．The VLDB Journal，2014，23（1）：77-102

[2] GETOOR L，MACHANAVAJJHALA A．Entity resolution: theory, practice & open challenges [J]．Proc VLDB Endow，2012，5（12）：2018-2019

[3] M LLER H，FREYTAG J-C. Problems, methods, and challenges in comprehensive data cleansing [M]. Professoren des Inst. Für Informatik. 2005.

[4] WIDYANTORO D H，IOERGER T R，YEN J. An incremental approach to building a cluster hierarchy [M]. Data Mining, 2002 ICDM 2003 Proceedings 2002 IEEE International Conference on. IEEE. 2002: 705-708.

[5] CHARIKAR M，CHEKURI C，FEDER T，et al. Incremental clustering and dynamic information retrieval [M]. Proceedings of the twenty-ninth annual ACM symposium on Theory of computing. ACM. 1997: 626-635.

[6] BENJELLOUN O，GARCIA-MOLINA H，MENESTRINA D，et al．Swoosh: a generic approach to entity resolution [J]．The VLDB Journal，2009，18（1）：255-276

[7] PHAM D T，DIMOV S S，NGUYEN C．An incremental K-means algorithm [J]．Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science，2004，218（7）：783-795

[8] MATHIEU C，SANKUR O，SCHUDY W．Online correlation clustering [J]．arXiv preprint arXiv:10010920，2010，12（3）：21-36

[9] GRUENHEID A，DONG X L，SRIVASTAVA D. Incremental Record Linkage [M]. Proceedings of the VLDB Endowment. Hangzhou, China; VLDB Endowment. 2014: 20-12.

[10] CAN F．Incremental clustering for dynamic information processing [J]．ACM Trans Inf Syst，1993，11（2）：143-164

[11] AGGARWAL C C，HAN J，WANG J，et al. A framework for clustering evolving data streams [M]. Proceedings of the 29th international conference on Very large data bases - Volume 29. Berlin, Germany; VLDB Endowment. 2003: 81-92.
[12] DEZ M A H，STOLFO S J．Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem [J]．Data Min Knowl Discov，1998，2（1）：9-37
[13] CHRISTEN P．A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication [J]．IEEE Trans on Knowl and Data Eng，2012，24（9）：1537-1555
[14] KOUDAS N，SARAWAGI S，SRIVASTAVA D. Record linkage: similarity measures and algorithms [M]. Proceedings of the 2006 ACM SIGMOD international conference on Management of data. ACM. 2006: 802-803.
[15] ELMAGARMID A K，IPEIROTIS P G，VERYKIOS V S．Duplicate Record Detection: A Survey [J]．IEEE Trans on Knowl and Data Eng，2007，19（1）：1-16
[16] SARAWAGI S，BHAMIDIPATY A. Interactive deduplication using active learning [M]. Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. Edmonton, Alberta, Canada; ACM. 2002: 269-278.
[17] WINKLER W E. Overview of record linkage and current research directions [M]. Bureau of the Census. Citeseer; BUREAU OF THE CENSUS. 2006.
[18] ARASU A，G M，#246，et al. On active learning of record matching packages [M]. Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. Indianapolis, Indiana, USA; ACM. 2010: 783-794.
[19] DOAN A，HALEVY A，IVES Z. Principles of data integration [M]. Elsevier. 2012.
[20] DRAISBACH U，NAUMANN F. On choosing thresholds for duplicate detection [M]. Proceedings of the 18th International Conference on Information Quality (ICIQ). 2013.
[21] GU L，BAXTER R. Decision models for record linkage [M]. Data Mining. Springer. 2006: 146-160.
[22] PETER C. Data Matching Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection [M]. springer. 2012.
[23] MONGE A E．Matching algorithms within a duplicate detection system [J]．IEEE Data Eng Bull，2000，23（4）：14-20
[24] NAUMANN F，HERSCHEL M. An Introduction to Duplicate Detection [M]. Morgan and Claypool Publishers. 2010: 92.
[25] NAUMANN F，HERSCHEL M．An introduction to duplicate detection [J]．Synthesis Lectures on Data Management，2010，2（1）：1-87
[26] HERN M A，#225，NDEZ，et al. The merge/purge problem for large databases [M]. Proceedings of the 1995 ACM SIGMOD international conference on Management of data. San Jose, California, USA; ACM. 1995: 127-138.
[27] MONGE A E，ELKAN C. The Field Matching Problem: Algorithms and Applications [M]. KDD. 1996: 267-270.
[28] BANSAL N，BLUM A，CHAWLA S．Correlation clustering [J]．Machine learning，2004，56（1-3）：89-113
[29] CHARIKAR M，GURUSWAMI V，WIRTH A．Clustering with qualitative information [J]．J Comput Syst Sci，2005，71（3）：360-383
[30] HASSANZADEH O，CHIANG F，LEE H C，et al．Framework for evaluating clustering algorithms in duplicate detection [J]．Proc VLDB Endow，2009，2（1）：1282-1293
[31] BENJELLOUN O，GARCIA-MOLINA H，MENESTRINA D，et al．Swoosh: a generic approach to entity resolution [J]．The VLDB Journal—The International Journal on Very Large Data Bases，2009，18（1）：255-276